TEXAS INSTRUMENTS
INCORPORATED
POST OFFICE BOX 10508 • LUBBOCK, TEXAS 79408
U.S. Consumer Products Group

File: Letters Sent

April 30, 1983

MCC Powers
2942 MacArthur Blvd.
Northbrook, IL 60062
Attn: John Jeffers

Dear Sir,

Enclosed you will find documentation and a listing of a BASIC program that uses RS-232 polling instead of using service requests. This program illustrates a useful technique that avoids the problems you have encountered using service requests with the RS-232 peripheral. Although the enclosed program is in BASIC, the technique applies equally well in assembly language. We have used it with success in several assembly language applications.

If you have a question regarding this example, please contact me at 214-997-2454.

Sincerly,

Bud Gerwig

Bud Gerwig
Texas Instruments
8390 LBJ Freeway
Dallas, TX 75243
214-997-2454

/lb
enclosure
cc: Ross Deem
[redacted]
Randy Marker

Fifty Years of Innovation

BASIC Polling of RS232 Peripheral

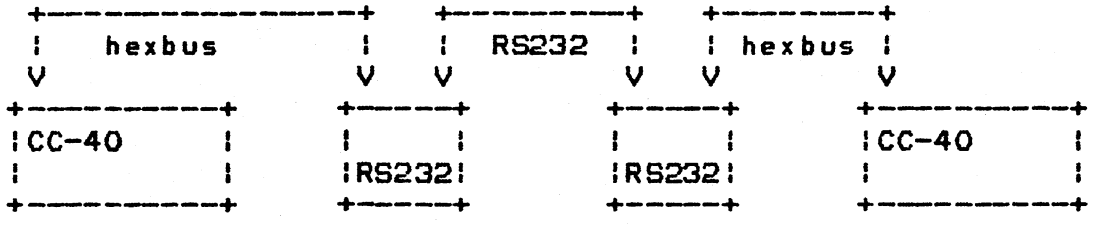
Texas Instruments, Inc.
April 30, 1984

SECTION 1

Polling The RS232

1.1 Polling

Polling the RS232 in the proper mode can be used to transfer data from the RS232 to the CC-40 in some cases. The example program in BASIC given in this paper uses the RS232 in a polling environment to transfer one character at a time between two RS232 peripherals. Shown below is the test setup that was used with this program.



note: RS232 cables pins 2,3 are swapped.
pin 2 -> pin 3 other end
pin 3 -> pin 2 other end

The program given below is entered in BOTH CC-40 units, and run. The program opens each RS232 in the same mode, allowing for both input and output, one character at a time. When a key is pressed on one unit, it is displayed in the left half of the display of that unit, and transmitted across the RS232 to the other unit. The unit then polls the RS232 for any data that it has. If any data comes back, it is displayed in the right half of the display. Else, if no data comes back, the program goes back to see if any keys were pressed. The basic effect of this program is to enable two-way communication on a character by character level through the RS232. Whatever is entered on one keyboard is echoed to that units display, and displayed on the other units display also. Given below is the documented code which performs this operation.

```
100 !=========================================
110 ! RS232 Polling Example
120 ! April 30, 1984
130 !=========================================
140 ! Open the peripheral.  Device number as
150 ! required.
160 OPEN #1,"20.B=300,R=N,P=O,C=N,E=N,T=C",UPDATE,VARIABLE 1
170 ! peripheral opened in following mode:
180 ! baud: 300
190 ! no carriage return
200 ! no echo
210 ! transfer is by character
220 ! opened for input and output (update)
230 ! no parity check
240 ! buffer is length 1 (character)
250 !-----------------------------------------
260 ! Initialized pointers and check for key
270 CR=1:OU=16
280 A$="":CALL KEY(K,S):IF S=0 THEN 430
290 ! Key is down, so display in left and
300 !  transmit to RS232.
310 A$=CHR$(K)
320 IF CR>15 THEN CR=1   ! check for wrap on display
330 PRINT #1,A$          ! transmit character
340 ! next statement clears half of display
350 IF K=13 THEN DISPLAY AT(1)SIZE(15);RPT$(" ",15);:CR=1:GOTO 430
360 IF K=7 THEN 430      ! ignore bells
370 DISPLAY AT(CR)SIZE(1);A$;  ! echo character to display
380 CR=CR+1              ! move to next position
390 !-----------------------------------------
400 ! Poll the RS232 for any data.  Returns
410 !  null data if none present.
420 !-----------------------------------------
430 LINPUT #1,B$:IF B$="" THEN 280 ! if none, go look for key
440 ! Got some data from the RS232.
450 IF ASC(B$)=13 THEN DISPLAY AT(16);" ":OU=16:GOTO 280
460 IF ASC(B$)=7 THEN DISPLAY BEEP:GOTO 280
470 !-----------------------------------------
480 ! Display data in right half of display
490 !-----------------------------------------
500 DISPLAY AT(OU)SIZE(1);B$;:OU=OU+1
510 IF OU>31 THEN OU =16 ! Wrap data on display
520 GOTO 280
```

1.2 RS232 Actions

When the RS232 is set up in a polling mode, such as given above, it operates in the following manner:

```
IF ANY DATA IS TO BE TRANSMITTED, SEND IT
WHEN A REQUEST FOR DATA COMES IN, THEN
  IF ANY DATA IS READY TO RETURN, SEND IT
  ELSE SEND A NULL LENGTH DATA
```

The peripheral sets up two data buffers internal to the RS232 when opened in UPDATE (input and output) mode. One buffer is set aside for incoming data from the CC40, and the other for incoming data from the RS232. When the CC40 sends data, it is placed in the proper buffer, and transmitted across the RS232, according to the options which were specified in the OPEN statement. When the CC40 requests data from the peripheral, a check is made on the RS232 data buffer to see if any data is there. If there is no data in the buffer, a null data field (0 bytes data) is immediately returned, otherwise, any data present is returned, up to the length of the CC40's data buffer. In the example above, this buffer is one character long so data is returned character by character.